Robotics and Automation, April 7-1
1986, San Francisco, CA

TIME OPTIMAL TRAJECTORY PLANNING FOR ROBOTIC MANIPULATORS WITH OBSTACLE AVOIDANCE: A CAD APPROACH

S. Dubowsky
M.A. Norris
Z. Shiller

Department of Mechanical Engineering
Massachusetts Institute of Technology
Cambridge, MA 02139

ABSTRACT

A method is presented which finds the minimum time motions for a manipulator between given end states. The method considers the full nonlinear manipulator dynamics, actuator saturation characteristics, and accounts for both the presence of obstacles in the work space and restrictions on the motions of the manipulator's joints. The method is computationally practical and has been implemented in a Computer Aided Design (CAD) software package, OPTARM II, which facilitates its use. Examples of its application to a six degree-of-freedom articulated manipulator, performing tasks in a typical environment, are presented. The results show that substantial improvements in system performance can be achieved with the technique.

INTRODUCTION

Robotic manipulators in current practice often do not perform their tasks in the minimum times possible because their motions are planned manually. Such planning generally fails to select the best paths or the maximum permissible speeds at all points along those paths. This results in lower productivity in industrial applications and reduced effectiveness in non-industrial uses, as in space. Manual planning of minimum time manipulator motions is very difficult because; manipulators have highly nonlinear dynamic characteristics, their actuator capabilities are usually functions of their dynamic state; and their motion times are complex functions of their paths.

Considerable research has been done to find an analytical solution to this problem. The first significant study used classical optimal control theory to obtain the solution for a simplified linearized manipulator model[1]. Later the nonlinear problem was solved numerically using optimal control methods, but the intensive computation required have restricted this approach to simple manipulator models[2]. These solutions did not include geometric constraints, such as obstacles. Other solutions typically involve substantial assumptions that limit their usefulness. Some have assumed "bang-bang" form solutions with a specifed number of switching points, but since the actual number of switching points is not known, a true optimal solution cannot be guaranteed[3,4]. Others applied dynamic-programming techniques to the non-linear problem by searching a tessellated state space for the optimal trajectory[7]. However this method assumes straight line segments and thereby neglects the important effects of path curvature. Computation time for such an approach can be a problem if it is applied to realistic systems. Others have attempted to develop approximate solutions to the problem to reduce the computational effort[6]. Research to find a practical solution to a major subproblem of the general optimal motion problem has met with considerable success. In this work the problem is limited to finding the manipulator's velocity profile along a prescribed path so that it is traversed in minimum time without violating the physical capabilities of the system, such as actuator saturation limits. Clearly, obstacle avoidance is not an issue in this problem. A rigorously optimal, computationally efficient algorithm has been developed for finding the minimum time motion of a manipulator along a prescribed path[7,8]. It considers the full nonlinear dynamics of the manipulator and permits actuator constraints to be expressed as complex functions of the system state. Subsequent research has shown the algorithm is practical for controlling general six degree-of-freedom manipulators, a useful design tool when used in the form of a highly interactive program, and that it can be extended to include such constraints as the maximum dynamic forces that an object being tolerate and the speeds which a manipulator can sustain without losing its grasp of the object[9,10]. Subsequently, other researchers have suggested essentially the same algorithm[11].

Recent research suggests that this algorithm may be used to find optimal paths[12]. Conceptually the algorithm could search various paths for the one with the lowest time. So far this approach has been applied only to either very simple problems or very special cases[13,14]. This paper shows that this approach can be made practical for realistic problems, and that it can be extended to consider the work space obstacles and manipulator joint motion limitations found in all realistic systems.

The method finds the minimum time motion of a manipulator between given end states, considers the full nonlinear dynamics of the manipulator, and actuator saturation characteristics (specified as functions of systems's dynamic state), and end effector and payload constraints. It permits the constraints imposed by obstacles and joint limits

to be represented in their natural coordinate frames by using a penalty function optimization approach. It handles these mixed constraint functions in a very computationally efficient and flexible manner. The method has been implemented in a Computer Aided Design (CAD) software package, called OPTARM II, with extensive interactive graphics capabilities which enhance its practical use. Examples are presented of its application to planning the motions of a six degree-of-freedom articulated manipulator performing tasks in a typical highly structured environment. Results show that substantial improvement in system performance can be achieved with the technique.

ANALYTICAL DEVELOPMENT

The Basic Time Optimal Control Algorithm

The basic time optimal control algorithm is derived in References 7 to 10. The algorithm obtains the open loop torques/forces for the time optimal motion of a general six degree-of-freedom manipulator along a prescribed path subject to actuator constraints (see Figure 1). The algorithm also provides optimal joint positions and velocities for closed loop control. The method is applicable to manipulators with rigid links for which the dynamic model and the joint coordinates can be defined for any point on the path.

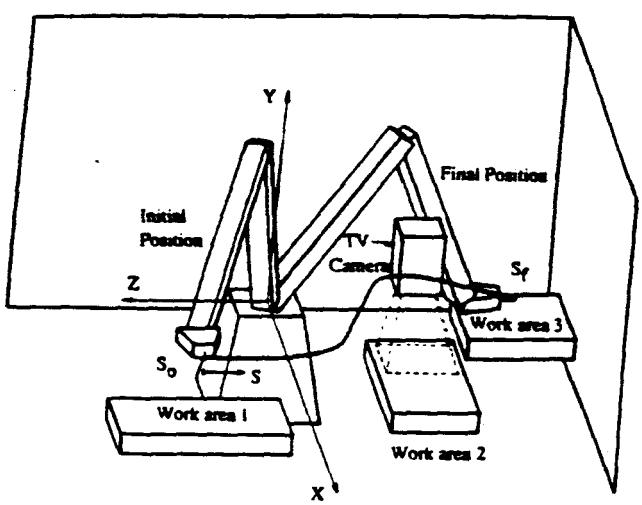

Fig. 1. A Six Degree-of-Freedom Manipulator in its Environment

Following the method in References 7 to 10, the dynamics equations of the manipulator are first written in terms of the joint position vector, $\underline{\theta}$, and its time derivatives $\dot{\underline{\theta}}$. The vector $\underline{\theta}$ is:

$$[\theta_1,\theta_2,\theta_3,\theta_4,\theta_5,\theta_6]^T$$

Since the path is a known function of S, the joint variables θ and $\dot{\theta}$ can be expressed in terms of the distance along the path, S, and the tip velocity and acceleration along the path, $\dot{S}$ and $\ddot{S}$, using the kinematic transformation from work-space coordinates, $\underline{P}$, to the manipulator joint angles, $\underline{\theta}$:

$$\underline{P}(S) = \underline{R}(\underline{\theta}) \qquad (1)$$

where $\underline{P}$ is $[x,y,z,\theta_1,\theta_2,\theta_3]^T$, the position and orientation of the end-effector. Differentiating Equation (1) twice with respect to time and solving for $\dot{\theta}$ and $\ddot{\theta}$ yields:

$$\begin{aligned}\dot{\underline{\theta}} &= R_\theta^{-1}\,\underline{P}_s\dot{S}\\ \ddot{\underline{\theta}} &= R_\theta^{-1}\{\underline{P}_s\ddot{S} + \underline{P}_{ss}\dot{S}^2 - (R_\theta^{-1}\underline{P}_s)^T R_{\theta\theta}(R_\theta^{-1}\underline{P}_s)\dot{S}^2\}\end{aligned} \qquad (2)$$

where the subscripts s and e denote partial derivatives with respect to the scalar S and vector $\underline{\theta}$, respectively. Using Equation set (2), the dynamics equations, written in terms of $\underline{\theta}$, $\dot{\underline{\theta}}$ and $\ddot{\underline{\theta}}$ can be transformed into:

$$\underline{m}(\underline{\theta})\ddot{S} + \underline{b}(\underline{\theta})\dot{S}^2 + \underline{G}(\underline{\theta}) = \underline{T} \qquad (3)$$

where m, b and G are functions of the manipulator's kinematic stucture, $R(\theta)$ and its derivatives, and the partial derivatives with respect to S. The vector $\underline{T}$ is composed of the actuator efforts. The ith actuator saturation limits, T_{imin} and T_{imax}, may be functions of $\underline{\theta}$ and $\dot{\underline{\theta}}$:

$$T_{imin}(\underline{\theta},\dot{\underline{\theta}}) \le T_i \le T_{imax}(\underline{\theta},\dot{\underline{\theta}}) \quad i = 1,2,\ldots,6 \qquad (4)$$

Reference 6 rigorously shows that the time to traverse the path from the given initial condition to the required final condition will be minimal if the manipulator's acceleration, $\ddot{S}$, is equal to either its maximum permissible value, $\ddot{S}_a$, or its minimum permissible value, $\ddot{S}_d$, at each point along the path. The values of $\ddot{S}_a$ and $\ddot{S}_d$ are obtained by solving the vector Equation (3) for $\ddot{S}$ successively, with the bounds on T given by Equation (4), to give six pairs of quadratic equations in S, for each point on the path:

$$\begin{aligned}\ddot{S}_{ai} &= (T_{imax} - b_i\dot{S}^2 - G_i)m_i\\ \ddot{S}_{di} &= (T_{imin} - b_i\dot{S}^2 - G_i)/m_i\end{aligned} \qquad i = 1,2,\ldots,6 \qquad (5)$$

The values of $\ddot{S}_{ai}$ and $\ddot{S}_{di}$ obtained from Equation (5) are the upper and lower bounds on a range of permissible accelerations defined by the capabilities of the ith actuator. Clearly, the permissible acceleration for the manipulator at each point on the path, S, is one that satisfies all actuator constraints. Hence it must lie in the intersections of all the ranges given by equation (5), $\ddot{S}_{di} < \ddot{S}i < \ddot{S}_{ai}$, Reference 10 shows that other physical constraints can be treated in a similar manner to actuator constraints. The size of the permissible acceleration range depends on the velocity $\dot{S}$. Generally, as the manipulator's speed increases, the range of the permissible accelerations aproaches zero (or $\ddot{S}_d = \ddot{S}_a$), at some value of the velocity, $\dot{S}_m$. For $\dot{S}$ greater than $\dot{S}_m$, no solution for the acceleration $\ddot{S}$ exists, which means that the manipulator is not capable of maintaining the path. Ploting $\dot{S}_m(S)$ versus S in the phase plane yields a velocity limit curve. If

the motion trajectory crosses this curve, one or more of actuator capabilites will be exceeded and the manipulator will leave its prescribed path.

Reference 10 presents a very efficient algorithm that solves the minimum time problem by finding the switching points between $\bar{S}_a(S,\dot{S})$ and $\bar{S}_i(S,\dot{S})$ such that the velocity $\dot{S}$ is always maximum, but does not cross the limit curve, $\dot{S}_a$, in the phase plane. The time to travel along the path is given by:

$$t_p = \int_{S_o}^{S_f} \frac{dS}{\dot{S}} \tag{6}$$

Clearly, the higher the velocity at each point along the path, the shorter the travelling time. The time obtained by conventional controllers is always longer than the optimal time obtained by this algorithm.

Path Optimization without Constraints

With the ability to find the minimum traveling time along a specified path, t_p, it is now possible to optimize the path to find the one with the smallest minimum time, t_a. The time t_p is calculated during the optimization, using Equation (6).

First, the path is represented by a finite set of n scalar parameters, a_1 to a_n, which can be written in vector form as:

$$\underline{a} = \{a_1, a_2, a_3, \ldots a_n\}^T \tag{7}$$

As discussed later, several methods for parameterizing manipulator paths were considered. There will be a single value of t_p for each path, or set of path parameters:

$$t_p = t_p(\underline{a}) \tag{8}$$

To find the optimal path, $t_p(a)$ is defined as the objective (or cost function) $J(a)$, which is to be minimized in the form of the unconstrained minimization problem:

$$\text{Minimize} \quad J(a) = t_p(\underline{a}) \tag{9}$$

A number of methods exist for the numerical solution of this problem. The procedure developed here is not restricted to a particular technique; and several were successfully used in this study, including exhaustive search techniques. The Pattern Search method proved to be the most effective for the majority of the problems treated here. This method is known for its ability to perform satisfactorily in problems with a wide range of characteristics[15]. A detailed description of these methods is beyond the scope of this paper.

Path Optimization with Obstacles and Joint Motion Constraint Using Penalty Functions.

The presence of work space obstacles and joint motion limitations add additional constraints to the optimization problem described above. These are important both because manipulator paths which come too close to other objects generally are unacceptable, and because the motions of manipulators are limited by practical design considerations that require their joints have only finite ranges of movements. The unconstrained optimization will often find its optimal path in these unacceptable regions. These constraints can be written in the form:

$$\begin{aligned} & g_j(\underline{a}) < 0 \qquad j = 1,2,\ldots,q \\ & \theta_i - \theta_{imin} > 0 \\ & \theta_i - \theta_{imax} < 0 \end{aligned} \qquad i = 1,2,\ldots,6 \tag{10}$$

where $g_j(a)$ are the obstacle constraint functions, q is the number of obstacles, and θ_{imin} and θ_{imax} are the joint limits on the ith joint.

The unconstrained optimization procedures, such as those described above, are not applicable to this problem. The use of constrained optimization techniques is complicated by the fact that these constraints are most easily expressed in different coordinates, workspace coordinates for obstacles and joint space coordinates for joint limits. Also, because of such factors as dynamic disturbances, it would be undesirable to have the manipulator move along a constraint. It would be more desireable to have the manipulator stay some distance from the constraint. This distance would ideally be selected, in part, based on the additional time it would require. In short, a planner might be willing to move closer to an object or a joint limit if that resulted in a very substantial time savings; otherwise he would most likely prefer to stay further away from it.

For these reasons, the penalty function optimization approach[16] was selected for this problem. In this method a weighted penalty for coming close to a constraint is added to the cost function, $t_p(a)$. Thus the constrained optimization problem is recast into the unconstrained form discussed above. The method can be used easily for constraints expressed in different coordinate systems, eliminating the need for computationally intensive coordinate transformations. Finally, the weighting factors provide flexibility in selecting the importance of maintaining a distance from a specific constraint. For the examples presented in this paper, the cost function was modified to:

$$J = t_p(\underline{a}) + \sum_{i=1}^{N} \frac{w_i}{d_i^m} + \sum_{i=1}^{M} \left\{ \frac{w_{\theta i}}{(\theta_i - \theta_{imin})^m} + \frac{w_{\theta i}}{(\theta_{imax} - \theta_i)^m} \right\} \tag{11}$$

where the w's are the weighting factors, the m's are positive integers, usually taken as 2, the d_i's are the minimal distance to the ith obstacle from the path, M is the number of joints in the manipulator, and N is the number of obstacles in the environment. OPTARM II uses a table of various general geometric shapes for computing the minimum distance to obstacles. More complex objects can be considered using solid modelling techniques.

Path Representation

As discussed earlier, to optimize the path it must be represented by a finite set of scalar parameters $[a_1, a_2, \ldots, a_n]$. The larger the number of variables used, the better the optimal solution because the optimization has more flexibility in finding the minimum time path. The disadvantage of using a larger number of parameters is that the computation time will be larger. The above path optimization method is not dependent upon any particular type of path representation, and a number of methods were considered in this study. The three that proved most effective were: Straight lines connected by circular arcs; perturbations about a straight line by a Fourier series; and splines. These have been integrated into the OPTARM II program.

Straight lines and circular arcs have the advantage of being easy to visualize; and many robotic manipulator program languages already use these functions for path representations. Fourier series representations, being distributed functions, were found to be efficient. Splines have the advantage that they allow easy manipulation of the path's shape while requiring only a limited number of parameters. They are also commonly used in CAD systems to model curved line segments. Hence they are compatible with the software used to model the complex geometry of obstacles and the manipulator itself[17,18]. The results presented below use Bezier splines and the use of this technique for path representation is presented below.

An individual Bezier spline, called a span, defines a curved line between points. Spans can be combined to form one large spline defining the entire curve over a large number of points, matching derivatives at connections between spans. The cubic form of the Bezier Spline is:

$$\underline{r} = (1-u)^3 \underline{r}_0 + 3u(1-u)^2 \underline{r}_1 + 3u^2(1-u)\underline{r}_2 + u^3 \underline{r}_3 \quad (12)$$

where $\underline{r}(u)$ is a point on the spline, an nx1 vector in n dimensional space. The $\underline{r}_i$ are the control points which define the spline, and u is a scalar variable, usually set to vary between 0 and 1. The points $\underline{r}_0$ through $\underline{r}_3$ are shown in Figure 2. The Bezier Spline is set up so that the spline passes through $\underline{r}_0$ and $\underline{r}_3$, and is tangent to the lines defined by $\underline{r}_0\,\underline{r}_1$ and $\underline{r}_2\,\underline{r}_3$. The spline's shape is modified by adjusting the positions of $\underline{r}_1$ and $\underline{r}_2$. The positions of the $\underline{r}_1$ and $\underline{r}_2$ vectors can be varied to obtain general geometric curves such as straight lines, parabolas, and circular arcs. Here, $\underline{r}$ is the the 6 x 1 vector $\underline{P}$, which defines the position and orientation of the manipulator's end effector on its path, and u is a normalized distance along the path, equivalent to S.

Writing Equation (12) in matrix form, the spline representation of the manipulator path becomes:

$$\underline{P}(u) = (1 \; u \; u^2 u^3) \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \underline{r}_0 \\ \underline{r}_1 \\ \underline{r}_2 \\ \underline{r}_3 \end{bmatrix} \quad (13)$$

which has the form:

$$\underline{P}(u) = \underline{U}\, M_b\, R \quad (14)$$

The partial derivatives of the path with respect to the distance along the path, S, required by Equation set (2), are obtained by differentiating the $\underline{U}$ matrix. For example:

$$\underline{U}_s = \frac{\partial \underline{U}}{\partial u} \cdot \frac{\partial u}{\partial S} \quad (15)$$

and thus:

$$\underline{P}_s = \underline{U}_s\, M_b\, R \quad (16)$$

These matrix representations of the spline paths and their derivatives are easily integrated into OPTARM II. The variables defining the locations of the control points compose the parameter vector, a, in the path optimization. It might also be noted that it is possible to permit the end point locations to "float" during the optimization, which essentially optimizes the location of the manipulator's work stations. The penalty function approach can be used to ensure that these locations are kept in appropriate places.

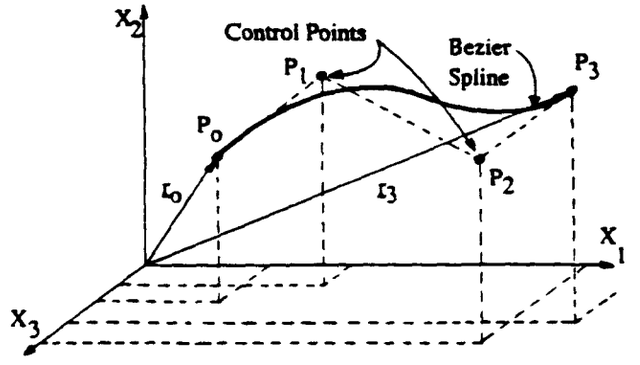

Fig. 2. A 3 Dimensional Bezier Spline Span.

EXAMPLES

OPTARM II integrates the basic optimal control algorithm with path representation, obstacles modeling, and path optimization subroutines to optimize automatically the motions of manipulators. Examples of the results obtained from OPTARM II are presented below. The figures are annotated versions of those produced by the OPTARM II graphics interface.

Optimization of the Motion Along a Specified Path.

Application of the basic method, without path optimization, can yield very substantial improvements in performance over conventional control methods. Here the manipulator moves from S_0 in Work Area 1, to S_f, in Work Area 3, along a path which has been chosen manually to be a "good path", avoiding the obstacles formed by Work Area 2 with its TV camera. This path is called an intuitive path. The work space coordinates of S_0 and S_f are (1.25,0.3,0.5) and (-0.5,-2.0,-1.25), respectively. The top view of this path and work place is shown in Figure 3. The path has been constructed in the typical industrial manner from straight line segments joined by circular arcs. The same path is shown in a perspective view in Figure 1.

In conventional manipulator control the tool point moves along its preprogrammed path from its initial point to its final point, at a specified constant speed. The system typically uses constant acceleration and deceleration at the end points. The values of the constant acceleration, deceleration and velocity are selected so that the manipulator does not leave the required path at any point. This means that at most points on the path the manipulator is moving slower than it can, and therefore its travel time is longer than the optimal.

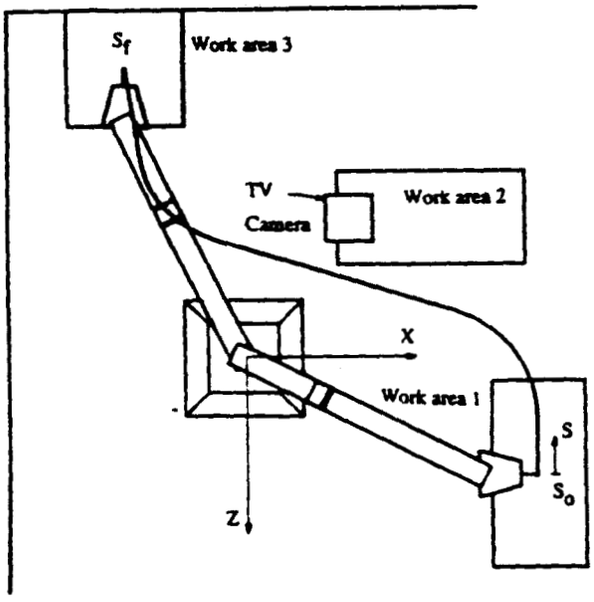

Fig. 3. Top View of A Manipulator on its Intuitive Path.

The conventional control trajectory in the phase plane, $S - \dot{S}$, is shown in Figure 4. As explained earlier, if the manipulator enters into the region above the limit curve its actuators will be saturated and it will leave its prescribed path. The conventional control is chosen to have the highest constant velocity that will not exceed this limit. This is done by just having the constant velocity section just pass under the lowest point of the limit curve. The time required to complete the move with the conventional control is calculated using Equation (6); for this case it is 1.28 seconds.

Also shown in Figure 4 is the optimal trajectory calcualted by OPTARM II for this path. It has three switching points and achieves higher velocities than the conventional control for a good portion of the path. There is a switch to deceleration so the trajectory just passes under the limit curve. At this point the trajectory switches to acceleration to gain as much speed as possible before it switches to the deceleration required to stop at the final position

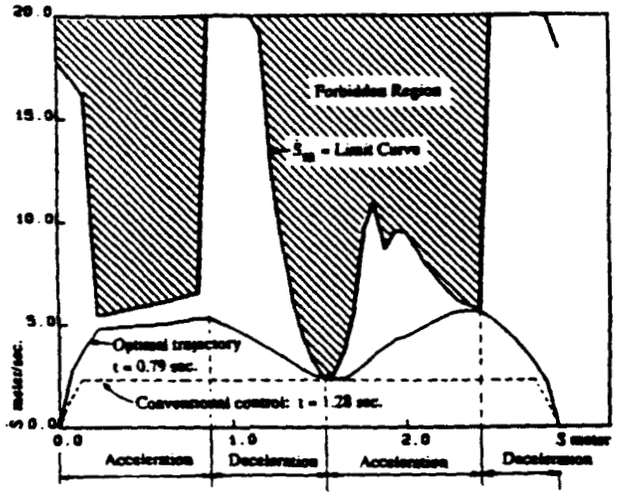

Fig. 4. Phase Plane Trajectories of Conventional and Optimal Motions for the Intuitive Path.

The optimal motion requires only 0.79 seconds. The conventional control takes 62% longer than the optimal. This is a very substantial improvement in performance. The computer graphics features of the program permits the display of the optimal dynamic motion of the manipulator in real time. The program also provides other important design and operating information, such as power consumption and actuator torque profiles. The torque profiles for this case are shown in Figure 5.

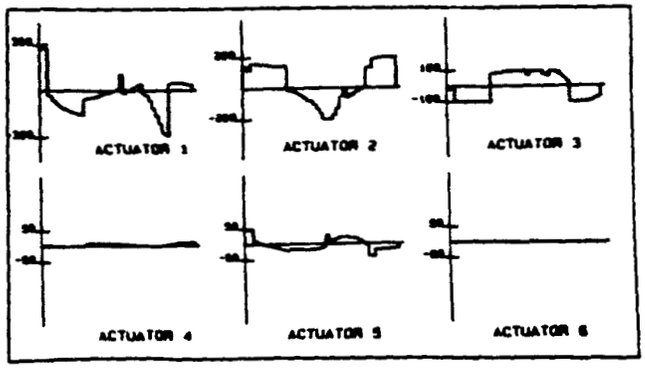

Fig. 5. OPTARM II Actuator Torque Profiles for the Intuitive Path, N-m vs S.

Path Optimization Without Constraints

The case described in the previous section was then optimized using OPTARM II. The end points were kept the same and the program optimized the path. The initial path for this optimization process is simply a straight line connecting the intial and final points, not a particularly sophisticated choice. The resulting optimal path is shown in Figure 6 along with the intuitive path. The time to move along this path is 0.47 seconds with two spline spans. This might be compared to .79 seconds for the optimal motion along the intuitive path, 68% longer. This optimal path passes over Work Area 2 and under the TV camera which may or may not be acceptable. It might be noted that this is less than one half of the time for the intuitive path with conventional control, a very significant improvement in performance.

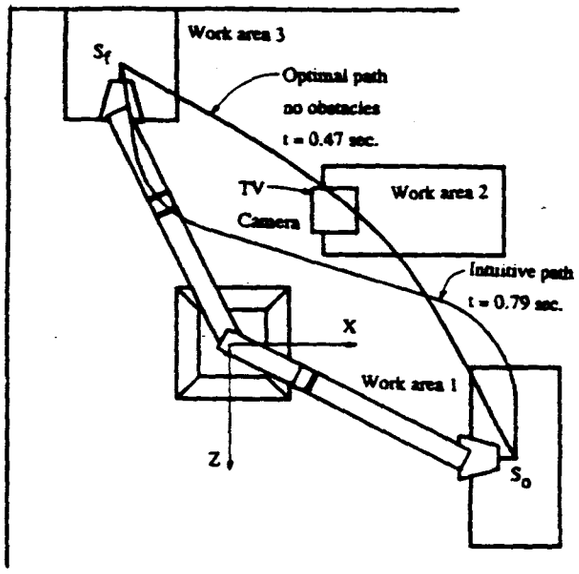

Fig. 6. Unconstrained Time Optimal Path (Without Regard for Obstacles)

The phase plane diagram for this optimized path is shown in Figure 7 along with its limit curve. Also shown in this figure is the optimized trajectory for the intuitive path repeated from Figure 4. In optimizing the path, the program has found a path with a significantly higher limit curve. This permits a higher trajectory in the phase plane which results in a shorter move time, refer to Equation (6). It might also be noted that while the intuitive path required three switches, the optimal path requires only one switch. This also serves to decrease the motion time.

The one-span spline optimal path, as might be expected, gave a slightly longer time of 0.48 seconds. The one-span spline requires somewhat less computation time. In any event, the computation times for this method do not represent a severe burden; the method is clearly intended as an off-line planning technique and therefore does not need to have real time computational speed. OPTARM II requires just a few seconds of computation time on a MicroVax II to optimize the motion along a given path. A path optimization will typically converge to within five percent of the true optimal with about 50 to 100 iterations, depending upon the initial path chosen to start the procedure and the restrictions on the range of acceptable paths to be considered. An effective optimization can usually be completed in minutes of CPU time. For tasks which are to be repeated a number of times, the increases in productivity would most certainly make this a cost effective procedure.

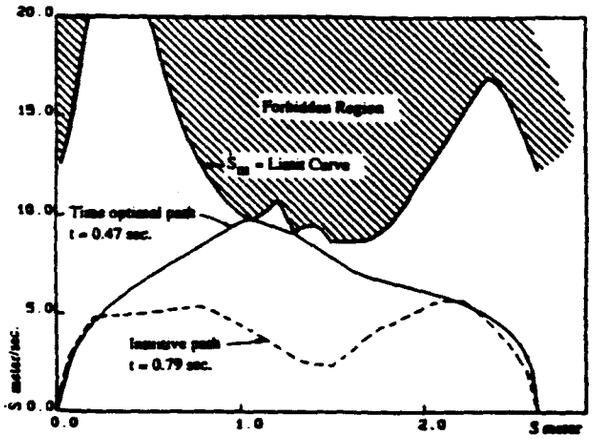

Fig. 7. Phase Plane Diagram for Unconstrained Optimal Path.

Path Optimization with Obstacles and Joint Motion Limits

Now consider the previous example, but where having the manipulator pass under the TV camera is unacceptable. In order to solve this problem the camera and the region directly below the camera were designated as obstacles to OPTARM II. The optimization procedure was repeated, again with an initial path of a straight line between the end points. The joint angle limits were sufficiently outside the range of prospective paths that they did not play a significant role in the optimization. The resultant path is shown in Figure 8 along with the unconstrainted optimal path. Figure 9 shows the final optimized phase plane diagram for this case. The limit curve is now lower in the region of the obstacle as the path is forced away from the camera by the penalty function. The travel time has increased from the unconstrained case value of 0.47 seconds to 0.53 seconds. This is a relatively small increase, and the tool point no longer passes through the camera field of view. It should be noted that this path is further away from the camera than the original intuitive optimal path and has a much shorter move time. The minimum distance to the obstacle could be increased or decreased simply by changing the weighting factor in the penalty function. Of course, there would be a corresponding increase or decrease in the motion time. The computation time for this case was virtually the same as the unconstrained optimal path case.

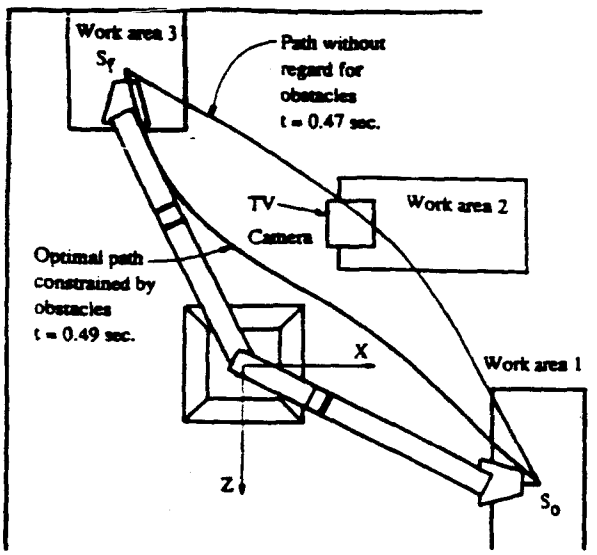

Fig. 8. Time Optimal Constrained and Unconstrained Paths.

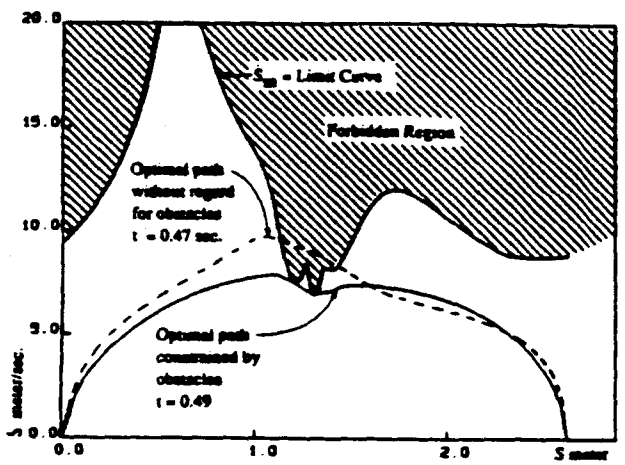

Fig. 9. Phase Plane Optimal Trajectories for Constrained and Unconstrained Paths.

SUMMARY AND CONCLUSIONS

In this paper a method is presented which finds the minimum time motions for a manipulator between given end states. It considers the full non-linear dynamics of the manipulator and the saturation characteristics of its actuators. Using a penalty function approach, it accounts for the presence of obstacles in the work space and restrictions on the motions of the manipulator's joints. It shows that the technique can also include such constraints as the maximum dynamic forces that can be tolerated by the object being manipulated, and the gripper. The method has proven to be computationally practical and has been implemented in a Computer Aided Design (CAD) software package, called OPTARM II. Examples of the use of the method with a six degree-of-freedom articulated manipulator, performing tasks in a typical highly structured environment, are presented. The results show that substantial improvements in system performance can be achieved with the technique.

ACKNOWLEDGMENTS

This research was supported by the Automation Branch of the NASA Langley Research Center under Grant NAG-1-489.

REFERENCES

1. Kahn M.E. and Roth B., "The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains," J. of Dynamic Sys., Meas. and Contr., Vol. 93, No. 3, Sept. 1971, pp. 164-172.
2. Meirovitch, A. and Bryson, A.E.,"Optimal Control of Systems with Hard Control Bounds," 1985 ACC, Boston, MA., pp. 1248-1252, June 1985.
3. Scheinman, V. and Roth, B., "On the Optimal Selection and Placement of Manipulators", Symp.on the Theory and Practice of Robots and Manipulators, Udine, Italy, June 1984.
4. Niv, M. and Auslander, D.M., "Optimal Control of a Robot with Obstacles," 1984 ACC, San Diego, CA., pp. 280-287, June 1984.
5. Sahar, G. and Hollerbach, J.M., "Planning of Minimum-Time Trajectories for Robot Arms," IEEE Intern. Conf. on Robotics and Auto. pp.751-758, St. Louis, Mo., Mar., 1985.
6. Shin, K.G. and McKay, N.D.,"Selection of Near-Minimum Time Geometric Paths for Robotic Manipulators," 1985 ACC, Boston, MA., pp. 346-355, June 1985.
7. Bobrow, J.E., Dubowsky, S., and Gibson, J.S., "On the Optimal Control of Robotic Manipulators with Actuators Constraints," 1983 ACC, San Francisco, CA, pp.782-787, June 1983.
8. Bobrow, J.E., Dubowsky, S., and Gibson, J.S., "Time-Optimal Control of Robotic Manipulators," Intern. J. of Robotics Res.,Vol.4, No. 3, 1985
9. Dubowsky, S. and Shiller, Z., "Optimal Dynamic Trajectories for Robotic Manipulators,"Symp. of the Theory and Practice of Robots and Manipulators, Udine, Italy, June 1984.
10. Shiller, Z. and Dubowsky, S., "On the Optimal Control of Robotic Manipulators with Actuator and End-Effector Constraints," IEEE Intern. Conf. on Robotics and Auto.,pp. 614-620, St. Louis, Mo., Mar.,1985.
11. Shin, K.G.and McKay, N.D.,"Open-Loop Minimum-Time Control of Mechanical Manipulators and its Application," 1984 ACC.,pp.1231-1236, June 1984
12. Shiller, Z. "Optimal Dynamic Trajectories and Modeling of Robotic Manipulators," MS Thesis, M.I.T., Cambridge, MA, June, 1984.
13. Rajan, V.T., "Minimum Time Trajectory Planning", 1985 IEEE Conf. on Robotics and Auto., pp. 759-764, St.Louis, Mo., March 1985.
14. Dubowsky, S. and Blubaugh, T.D., "Time Optimal Robotic Manipulator Motions and Work Places for Point to Point Tasks," 24th IEEE CDC, Fort Lauderdale, FL, Dec., 1985.
15. Bronson, R., "A Modified Pattern Search,"IEEE Trans. on Auto. Cont.,Vol.25, No.2, April 1980.
16. Walsh, G.R., Methods of Optimization, John Wiley & Sons, London, 1975, pp 148-150.
17. Bezier, P., "UNISURF System: Principles, Program, Language,"1973 PROLOMAT Conf., Budapest, 1973.
18. Faux, I.D. and Pratt, M.J., Computational Geometry for Design and Manufacture, Ellis Horwood Limited, New York, 1981.